Gossip Communities: Collaborative Filtering Through Peer-to-Peer Overlays*
(extended abstract)

R. Baraglia¹, P. Dazzi¹, M. Mordacchini¹, R. Perego¹, and L. Ricci²

¹ ISTI-CNR, Pisa, Italy
{r.baraglia, p.dazzi, m.mordacchini, r.perego}@isti.cnr.it
² Dept. Comp. Science, University of Pisa, Italy
ricci@di.unipi.it

Abstract. Gossip-based Peer-to-Peer protocols proved to be very efficient for supporting dynamic and complex information exchange among distributed peers. They are useful for building and maintaining the network topology itself as well as to support a pervasive diffusion of the information injected into the network. In this paper, we propose the general architecture of a system that tries to exploit the collaborative exchange of information between peers in order to build a system able to gather similar users and spread useful suggestions among them.

1 Introduction

In this paper, we propose the general architecture of a system that tries to exploit the collaborative exchange of information between peers in order to build a system able to gather similar users and spread useful suggestions among them. More precisely, we wish to push further the idea of exploiting collaboratively built recommender mechanisms, based on interest clustering and obtained through interactions among users. We propose to couple the two systems by exploiting gossip-style P2P overlay networks in order to ease the gathering of users with similar interests and then use the links established so far to spread recommendations among peers. The aim that we pursue is twofold. On one hand we want to build a flexible, adaptive system that allow the creation of communities of interests among users in a decentralized, distributed way. P2P approaches (and the gossip-based ones, in particular) scale well to large numbers of peers and deal gracefully with system dynamism, whereas centralized systems need expensive and complex techniques to ensure continuous operation under node and link failures. Moreover, the service is implemented through the collaboration of the peers without needing any centralized authority that would store all profiles and ratings of users and provide centralized-controlled recommendations.

* The authors acknowledge the support of Project FP6-033576, Building and Promoting a Linux-based Operating System to Support Virtual Organizations for Next Generation Grids (2006-2010), and the CNR program “Ricerca a tema libero”. The authors would like also to thank Matteo Fulgeri, who conducted several experiments.
On the other hand, we want to exploit such communities not only for sharing the knowledge about interesting items inside them, but also to overcome some traditional problems of recommender systems. In particular, one of the most common problems concerns the ability to recommend new items. In the system we are proposing, each neighbor of a peer $P$ will push recommendations to it about items that it believes might be of potential interest for $P$. This decision is taken locally, when a neighbor selects or knows from its connections of the existence of a new item whose characteristics are related with one or more of its communities. It can then suggest this item to $P$ and its other neighbors of all the related communities. This mechanism would allow a more efficient and rapid diffusion of the information. The remain of this paper is organized as follows: in Sec. 2 we revise the literature about the subject of this paper; in Sec. 3 we present the architecture of the proposed system, while in Sec. 4 we give an experimental evaluation of our approach. In Sec. 5, conclusions and possible further exploitations of this work are examined.

## 2 Related Work

The existence of a correlation of interests amongst a group of distributed users has been leveraged in a variety of contexts and for designing or enhancing various distributed systems. For peer-to-peer file sharing systems that include file search facilities (e.g., Gnutella, eMule, etc.), a sound approach to increase recall and precision of the search is to group users based on their past search history or based on their current cache content [1, 2]. Another potential use of interest clustering is to form groups of peers that are likely to be interested in the same content in the future, hence forming groups of subscribers in a content-based publish-subscribe system [3]. Moreover, interest correlation can be used to help bootstrapping and self-organization of dissemination structures such as network-delay-aware trees for RSS dissemination [4]. The correlation between the users past and present accesses has been used for user-centric ranking. In order to improve the personalization of search results, the most probable expectations of users are determined using their search histories stored on a centralized server [5, 6]. Nevertheless, the correlation between users with similar search histories is not leveraged to improve the quality of result personalization, hence making the approach sound only for users with sufficiently long search histories. An alternative class of clustering search engines uses semantic information in order to cluster results according to the general domain they belong in (and not as in our approach to cluster users based on their interests). This can be seen as a centralized, user-agnostic approach to improve user experience. The clustering amongst data elements is derived from their vocabulary. It presents the user with results along different interest domains and can help the user to disambiguate these results from a query that may cover several domains, e.g., the query word apple can relate to both food/fruits and computers domains. Examples of such systems are EigenCluster [7], SnakeT [8] or TermRank [9]. Nonetheless, these systems simply modify the presentation of results so that the user decides herself
in which domain the interesting results may fall these results are not in any way automatically tailored to her expectations. They do not also consider the clustering of interest amongst users, but only the clustering in content amongst the data. Other approaches cluster users on the basis of similarity between their semantics profile. Approaches of this kind of systems includes GridVine [10], the semantic overlay networks [11] and p2pDating [12]. They build a semantic P2P overlay infrastructure that relies on a logical layer storing data. They make use of heterogeneous but semantically related information sources whereas our approach does not rely on any kind of semantic interpretation. It, in principle, enables a broader exploitation of more heterogeneous data sources. Related with our proposal is Tribler[13], a P2P television recommender system. In contrast with our approach, neighbor lists can be directly filled in by the user herself using an interface. No topology or affinity property is considered. We propose a gossip system that construct and maintain in rest groups of dynamic users based on their past activities, without needing their direct intervention.

3 Proposed approach

In this section we present the general principles for the construction and use of a network of peers based on shared interest. A representation of a network of this kind is given in Fig.1. Links between peers are established when they have been interested in the same content in the past. In general, they are considered to potentially show interests for the same content in the future. Thus, peers collaboratively exchange useful recommendations among themselves. In order to group similar users, the protocol works by the means of a clustering algorithm. First, each peer decides, independently, which are the peers it is linked with. These one-to-one relationships are chosen on the basis of an interest-based distance, amongst the peer it encounters. Every time it gets in touch with a new peer, it can learn of the existence of new potential neighbors (i.e., similar users) and then communicate with them. Finally, it can learn of other, potentially better neighbors. When this process is stabilized, a peer may consider its neighborhood as the representative of a community of a shared interest. An important point here is that a peer is characterized by multiple interests. Hence, the process depicted
above is conducted separately for each of the interests of a peer. Connections are established and maintained separately for every distinct interest. Thus, we have a set of virtual different overlays, where each peer participates in as many groups is required to cover its interests. The situation is showed in Fig.2. In order to achieve this organization, each peer start a different stream of messages for each of its distinct interests.

3.1 Users Profile

We want to have profiles of peers created using the users’ interests. They can be represented by recently accessed resources, purchased items, visited pages, etc. Such gathered information has to be considered in a proper way, since it forms the basis over which the whole network will be built up. Generally speaking, let $\mathcal{S}$ be the set of items of the system and let $\mathcal{S}_p \subseteq \mathcal{S}$ be the subset of items belonging to a peer $p$. We consider that the profile $\pi$ of $p$ can be defined as

$$\pi_p = \{(i, C(i), R(i)) | i \in \mathcal{S}_p\}$$

where $i$ is an item of the set of $\mathcal{S}_p$, $C(i)$ is the content associated with $i$ (potentially void) and $R(i)$ is the rating given by $p$ to $i$. Moreover, $p$ has a set $I^p = \{I^p_1, \ldots, I^p_k\}$ of interests. Each of the items in $\pi_p$ may be associated with an interest $I^p_j$. We can then represent $\pi_p$ in the following way:

$$\pi_p = \bigcup_{j=1,\ldots,k} \pi_p(I^p_j)$$

where $\pi_p(I^p_j)$ is the set of items associated with the interest $I^p_j$. For performing this association, we assume each peer has a function $\gamma$ that given an item belonging to $\mathcal{S}$ decides the interest it should be associated with. More formally:

$$\gamma_p(i) = I^p_j \text{ with } i \in \mathcal{S}_p$$

Note that the set $I^p$ is specific for each distinct peer $p$. We do not assume any globally known labeling, categorization or subdivision of the objects in $\mathcal{S}$. Each peer organizes its own subdivision of $\mathcal{S}_p$ in the interests of $I^p$. It can then confront its objects divided per interest with the sets of the other peers it will be put in contact. Given two peers $p_1$ and $p_2$, $p_1$ will consider its local interest $I^p_1$ similar to the interest $I^p_2$ if it contains the most similar set of items among the other sets in $I^p_2$ with respect to the items in $I^p_1$. Thus, having a suitable method to describe each user interests coded in the peer profiles, we have to put attention on choosing a proper similarity function $\text{sim} : \Pi^2 \rightarrow \mathbb{R}$ to compare profiles, where $\Pi$ is the set of all possible profiles. This is a particular relevant point, since this function determines the relationships between peers on the basis of their interests. If each different interest is determined by different type of features, different similarity measures could be used to evaluate peers proximities with respect to each interest. Several measures can be exploited to this end. For instance, one common way is to use a metric that takes into account
the size of each profile, such as the Jaccard similarity, that has proven to be an effective similarity measure [1, 4]. Given two peers $p_1$ and $p_2$ and two interests $I_{s}^{p_1}$ and $I_{t}^{p_2}$, the similarity can be computed as

$$sim(p_1, p_2) = \frac{|\pi_{p_1}(I_{s}^{p_1}) \cap \pi_{p_2}(I_{t}^{p_2})|}{|\pi_{p_1}(I_{s}^{p_1}) \cup \pi_{p_2}(I_{t}^{p_2})|}$$

3.2 Creation of Interest Communities

As soon as each peer is able to compute its interest-based distance to any other peer, its objective is to group with other peers that have close-by interests, in order to form the basis for interests communities. This process is done in a self-organizing and completely decentralized manner, using a gossip-style communication. Each peer knows a set of other peers, namely its interest neighbors, and tries periodically to choose new such neighbors that are closer to its interest than the previous ones. This is simply done by learning about new peers from some other peer, then retrieving their profile, and finally choosing the $C$ nearest neighbors in the union of present and potential neighbors. When a peer $p$ enters the network, it is put in contact with one or more peers already taking part in the interest-proximity network. They use the profile similarity function to compute how similar they are. They consider each interest in the $I$ of $\pi_{p}$ separately and they confront it with their own. Moreover, the peers contacted by $p$ use the same similarity function to determine which are, among their neighbors, the most similar to $p$ and route the join request of $p$ toward them. All the peers that receive that request will react using the same protocol described above. All the interactions are shown in Algorithms 1 and 2. This mechanism will lead $p$ to learn the existence of the most similar nodes in the network and allow it to connect with them. In doing this process, the involved peers can only use their local knowledge to compare their respective profiles. Once the process is stabilized, $p$ can consider its neighbors as the representatives of a personal community of “friends” from which request and to which forward recommendations. Thus, the gossip protocol provide the basis for classical recommender systems in forming the set of similar users. This is done distributively and adaptively and the gossip protocol ensure a robust and constant maintenance over time. Recommendations can then be requested by $p$ to its neighborhood and it can forward the newly items it discovered to its neighbors using Algorithms 3 and 4.

4 Preliminary results

In order to evaluate the envisioned architecture and the algorithms presented in this paper, with respect to their ability to build a peer-to-peer system able to group users sharing common interests in a totally decentralized way, we developed a prototype implementing the gossip-based peer-to-peer protocols described by the Algorithms 1,2. The prototype has been developed and tested using the Overlay Weaver [14] peer-to-peer framework. The data used for the experiments
Algorithm 1
Let \( CR(P') \) be a connection request from another peer \( P' \)
Let \( \text{NewPeers} = \emptyset \)
if \( \text{Sim}(P, P') \geq \min_{P_i \in N(P)} \text{Sim}(P, P_i) \) then
Accept \( CR(P') \)
for all \( P_i \in N(P) \) do
if \( \text{Sim}(P_i, P') \geq \theta \) then
add \( P_i \) to \( \text{NewPeers} \)
end if
end for
add \( P' \) to \( N(P) \)
send \( \text{NewPeers} \) to \( P' \)
else
refuse \( CR(P') \)
end if

Algorithm 2
Let \( N(P) \) be the set of \( P' \)'s actual neighbors
for all \( P_i \in N(P) \) do
Get from \( P_i \) a set \( \text{NewPeers} \) from its neighborhood
for all \( P' \in \text{NewPeers} \) do
if \( P' \not\in N(P) \) then
connect with \( P' \)
if \( \text{Sim}(P, P') \geq \min_{P_j \in N(P)} \text{Sim}(P, P_j) \) then
add \( P' \) to \( N(P) \)
end if
end if
end for
end for

Algorithm 3
Let \( N_p(I_j) \) be the set of \( P' \)'s neighbors for the interest \( I_j \)
Receive a recommendation request from \( p' \in N_p(I_j) \)
for all \( i \in \pi_p(I_j) \) do
if \( \text{Sim}(p', i) \geq \theta \) then
recommend \( i \) to \( p' \)
end if
end for

Algorithm 4
Know about a new item \( h \)
Let \( I_i \) be the interest \( h \) is related to
Let \( N_p(I_i) \) be the neighborhood of peers interested in \( I_i \)
for all \( p' \in N_p(I_i) \) do
if \( \text{Sim}(p', h) \geq \theta \) then
recommend \( h \) to \( p' \)
end if
end for

Table 1: Active and passive threads and pull and push recommender algorithms comes from the MovieLens Dataset [15]. The data we used consists of 1 million ratings for 3900 movies by 6040 users. Since each user only rates a few movies, the data matrix is very sparse. This makes this dataset a good benchmark for our approach because the construction of communities with a sparse dataset is a real hard task. In the current prototype each user profile contains information about the movies seen by the respective user. Each peer stores the information about its neighborhood: the addresses and the profiles of its neighbors. Using this dataset we performed the several experiments varying: (i) the functions used for selecting both the peers with which to communicate and the best peers to send among the ones in the neighborhood; (ii) the neighborhood size; (iii) the number of peers in the network; (iv) the number of gossip-protocol iterations performed before measuring the similarity among a peer and its neighborhood. Regarding the points (i) and (ii), we implemented three different gossip protocols in the simulator: Cyclon, Vicinity [16] and Twinfinder. All of them have a behavior compliant with the Algorithms 1 and 2. Cyclon and Vicinity [16] are well known Algorithms, whereas Twinfinder is a customized version of Vicinity, we conceived, where the sender transfer to each neighbor a subset of the receiver’s most similar profiles. Each protocol has been tested under several different conditions, varying the maximum number of neighbors a peer can store (in the range [1-20]) and the number of nodes in the networks (in the range [1000-3000]). The results we achieved are depicted in Figure 3. It shows the comparison among the results provided by the three protocols when the network is made of 2000
peers, namely the half-way in the range [1000-3000]. It is easy to see that the two protocols, Vicinity and Twinfinder, exploiting the peer profile similarity in the peer selection process achieve better results. Those experiments gave us a further confirmation that the gossip protocols, and our Twinfinder in particular, are suitable solutions to build links among similar peers, hence for building up Interest Communities. Then, we decided to perform experiments for measuring the potential ability of gossip protocols for providing recommendations inside the Interest Communities. In order to do it we defined the “Coverage” measure. Given the set of movie genres liked by a user, we defined her Coverage as the percentage of user genres potentially recommendable to her by its neighborhood. In this case, we used as a simple function \( \gamma_p \) the distinction between genres of the movies in the dataset. Figure ?? shows in the two subfigures contained, respectively, the Coverage achieved by the three protocols and the number of peer interactions with which that coverage is obtained. Also in this case the similarity based protocols, and Twinfinder in particular, achieved better results both in terms of Coverage and in terms of Coverage speed. The algorithm convergence speed measurement is useful for evaluating the amount of time (or cycles) required by the gossip protocol to find a good set of neighbors.

5 Conclusion

The focus of this paper is on addressing the problem of clustering users in a purely decentralized way. This is particularly useful for enabling an automated creation of communities made from users sharing common interests. In this pa-
per we presented the overall architecture of a gossip-based peer-to-peer system exploiting collaboratively built search mechanisms.

References